SML HW2

Simran Goindani

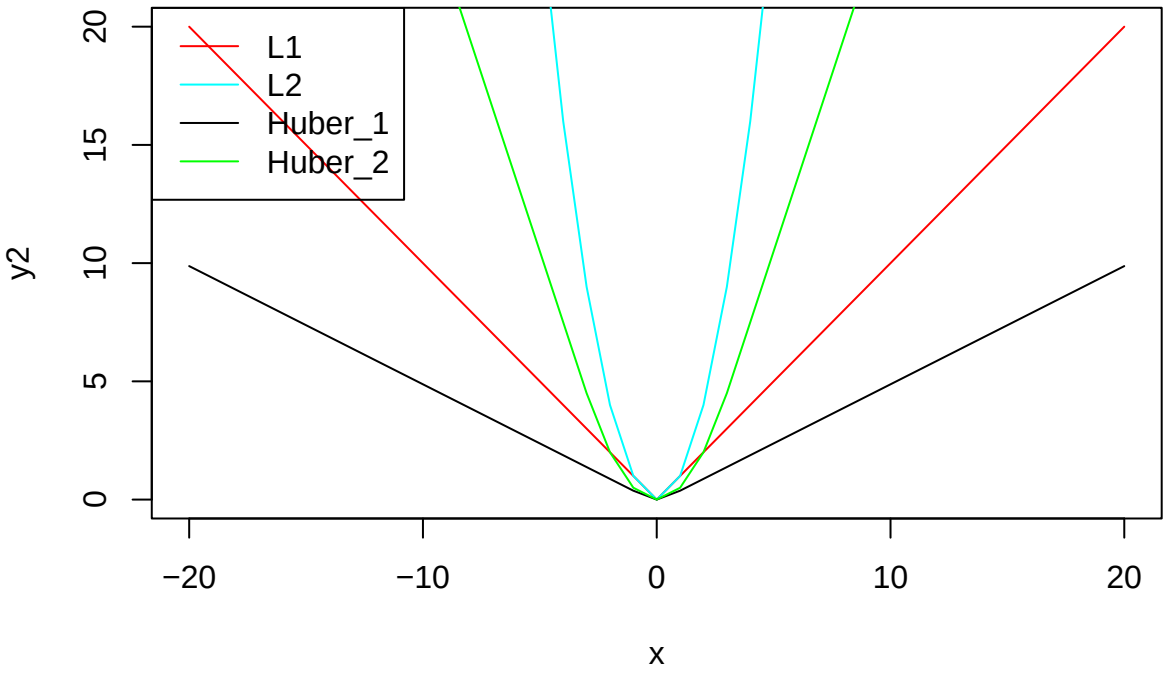
3/3/2022

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
set.seed(1)
x= c(-20:20)
y1 = x**2
y2=  abs(x)
delta1 = 0.5
delta2 = 3
mse = 0.5 * (x**2)
mae1 = delta1 * (abs(x) - 0.5*delta1)
mae2 = delta2 * (abs(x) - 0.5*delta2)
huber_1 = ifelse( abs(x) <= delta1, mse, mae1)
huber_2 = ifelse( abs(x) <= delta2, mse, mae2)


plot(x,y2,col = "red",type = "l")

lines(x,y1,col = "cyan")

lines(x,huber_1,col = "black")

lines(x,huber_2,col = "green")

legend("topleft",legend=c("L1", "L2","Huber_1","Huber_2"),
       col=c("red", "cyan","black","green"),lty =1)
```



Ans 3a: Advantage of MSE(L2) is that the trained model has no outlier predictions with huge errors,since MSE penalizes these errors by squaring it.

Disadvantage: If the model predicts a single bad prediction,then MSE magnifies it and it is alarming if there are few outliers

Advantage of MAE(L1) is that it doesn't magnify any outliers, weighing all of the errors on the linear scale.

Disadvantage: This loss function fails for outlier detection.

Advantage of Huber loss is that it is the best of MSE and MAE and creates a balance.As we can see the graph huber_2 is in between mae and mse

Disadvantage: Though it works well but we have to determine delta values correctly,as we can in the graph,huber_2 works better than huber_1.

```
# Gradient Descent
gradient_descent <- function(X,y,alpha,iterations,type){
  n = length(y)
  m <- runif(1, 0, 1)
  c <- runif(1, 0, 1)
  iter = 0
  hypothesis = m*X+c
  while (iter < iterations){

    if (type == "Quadratic"){
      loss = hypothesis - y
    }
    else if(type == "mean_absolute"){
      loss = hypothesis - y
      loss[loss>0] = 1
      loss[loss<0] = -1
    }
    else if (type == "huber"){
      delta = 2
```

```
      huber1 <- function(d, delta=2) ifelse (abs(d) <= a, 0.5*(d^2), a*abs(d)-0.5*d^2)
      loss = hypothesis - y
      loss[abs(loss) <= delta] = loss[abs(loss) <= delta]
      loss[abs(loss) > delta] = delta * sign(loss[abs(loss) > delta])
    }
    gradient = sum(loss*X)
    m_new <- m - alpha * ((1 / n) * gradient)
    c_new <- c - alpha *((1 / n) * sum(loss))
    m <- m_new
    c <- c_new
    hypothesis = m*X+c
    iter = iter +1
    }
  #print(paste("Optimal intercept:", c, "Optimal slope:", m))
  para = c(m,c)
  return(para)
}
```

```
#Stochastic Gradient Descent
Stoc_gradient_descent <- function(X,y,alpha,iterations,type){
  n = length(y)
  m <- runif(1, 0, 1)
  c <- runif(1, 0, 1)
  iter = 0

  while (iter<iterations){

    iter = iter+1
    j=0
    while (j < n){
      hypothesis = m*X+c
      yhat = m * X + c

      if (type == "Quadratic"){
        loss = hypothesis - y
      }
      else if(type == "mean_absolute"){
        loss = hypothesis - y
        loss[loss>0] = 1
        loss[loss<0] = -1
      }
      else if (type == "huber"){
        delta = 2
        huber1 <- function(x, delta=2) ifelse (abs(x) <= a, 0.5*(x^2), a*abs(x)-0.5*a^2)
        loss = hypothesis - y
        loss[abs(loss) <= delta] = loss[abs(loss) <= delta]
        loss[abs(loss) > delta] = delta * sign(loss[abs(loss) > delta])
      }
      gradient = sum(X*loss)
      m_new <- m - alpha * ((1 / n) * gradient)
      c_new <- c - alpha * ((1 / n) * (sum(loss)))
      m <- m_new
      c <- c_new
      j = j+1
```

```
    }

  }
  #print(paste("Optimal intercept:", c, "Optimal slope:", m))
  para = c(m,c)
  return(para)
}
```

```
library(pracma)
analytical_solution <- function(x,y){
   theta2=solve((t(X)%*%X))%*%t(X)%*%y
   return(theta2)
}
```

```
# Ans 4a

set.seed(1)
x = matrix(runif(50,-20,20))
e = rnorm(50,0,4)
X =  cbind(1,x)
y = 3 + 2 * x +e


print("Output of Analytical Solution :")
```

```
## [1] "Output of Analytical Solution :"
```

```
para1 = analytical_solution(X,y)
print(paste("Optimal Intercept: ",para1[1][1],"Optimal slope: ",para1[2][1]))
```

```
## [1] "Optimal Intercept:  3.38644304659435 Optimal slope:  2.00677892557822"
```

```
print("Output of batch gradient descent :")
```

```
## [1] "Output of batch gradient descent :"
```

```
para2 = gradient_descent(x,y,0.01,1000,"Quadratic")
print(paste("Optimal Intercept: ",para2[2][1],"Optimal slope: ",para2[1][1]))
```

```
## [1] "Optimal Intercept:  3.38630237975318 Optimal slope:  2.00678049401387"
```

```
print("Output of stochastic gradient descent :")
```

```
## [1] "Output of stochastic gradient descent :"
```

```
para3 = Stoc_gradient_descent(x,y,0.01,1000,"Quadratic")
print(paste("Optimal Intercept: ",para3[2][1],"Optimal slope: ",para3[1][1]))
```

```
## [1] "Optimal Intercept:  3.38644304659434 Optimal slope:  2.00677892557822"
```

```
# Ans 4b

ana_list = c()
batch_list = c()
stoc_list = c()
for (i in 1:1000){
  x = matrix(runif(50,-20,20))
  e = rnorm(50,0,4)
```

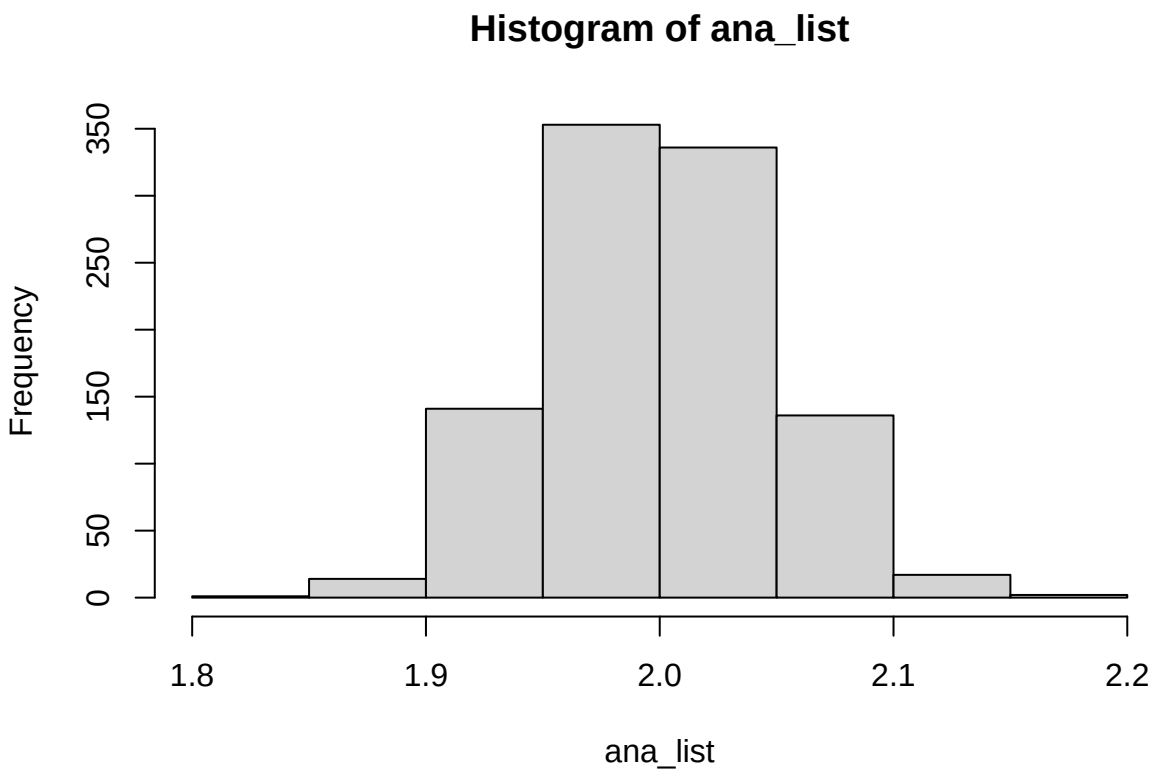
```
  y = 3 + 2 * x + e
  X = cbind(1,x)
  ana_list = append(ana_list,analytical_solution(X,y)[2])
  batch_list = append(batch_list,gradient_descent(x,y,0.01,1000,"Quadratic")[1])
  stoc_list = append(stoc_list,Stoc_gradient_descent(x,y,0.01,1000,"Quadratic")[1])
}
print("done")
```

```
## [1] "done"
```

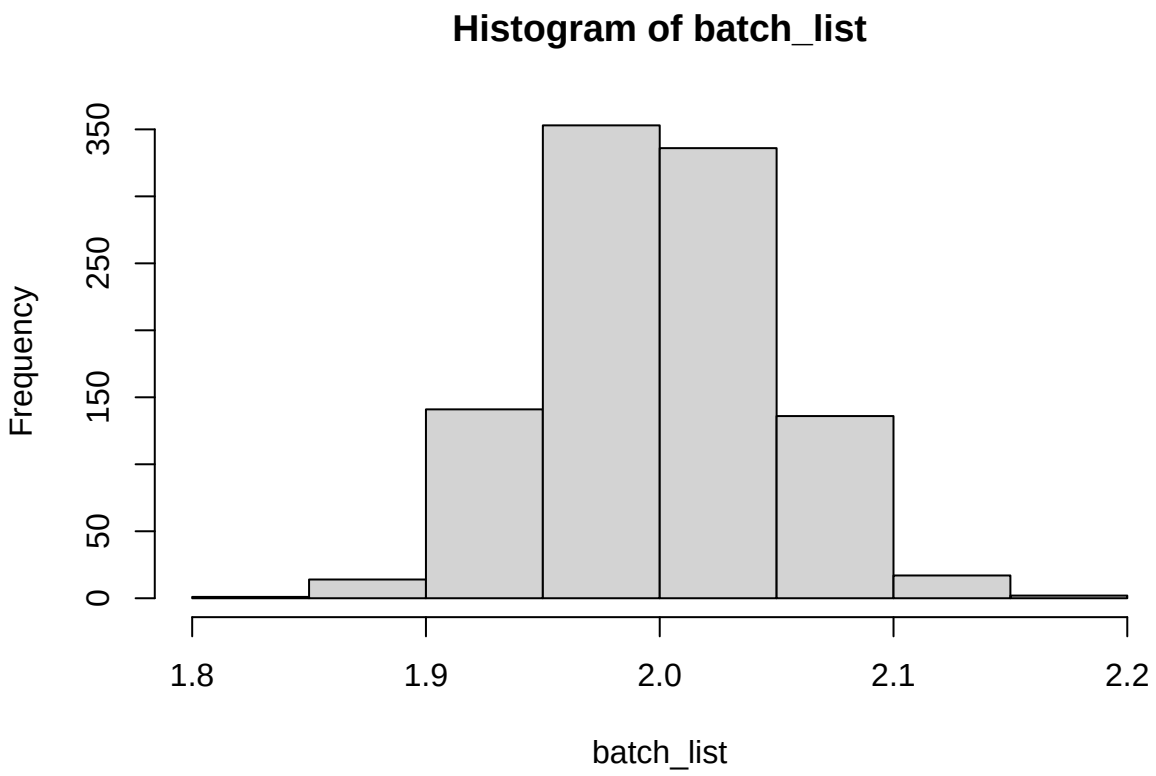
```
length(batch_list)
```

```
## [1] 1000
```

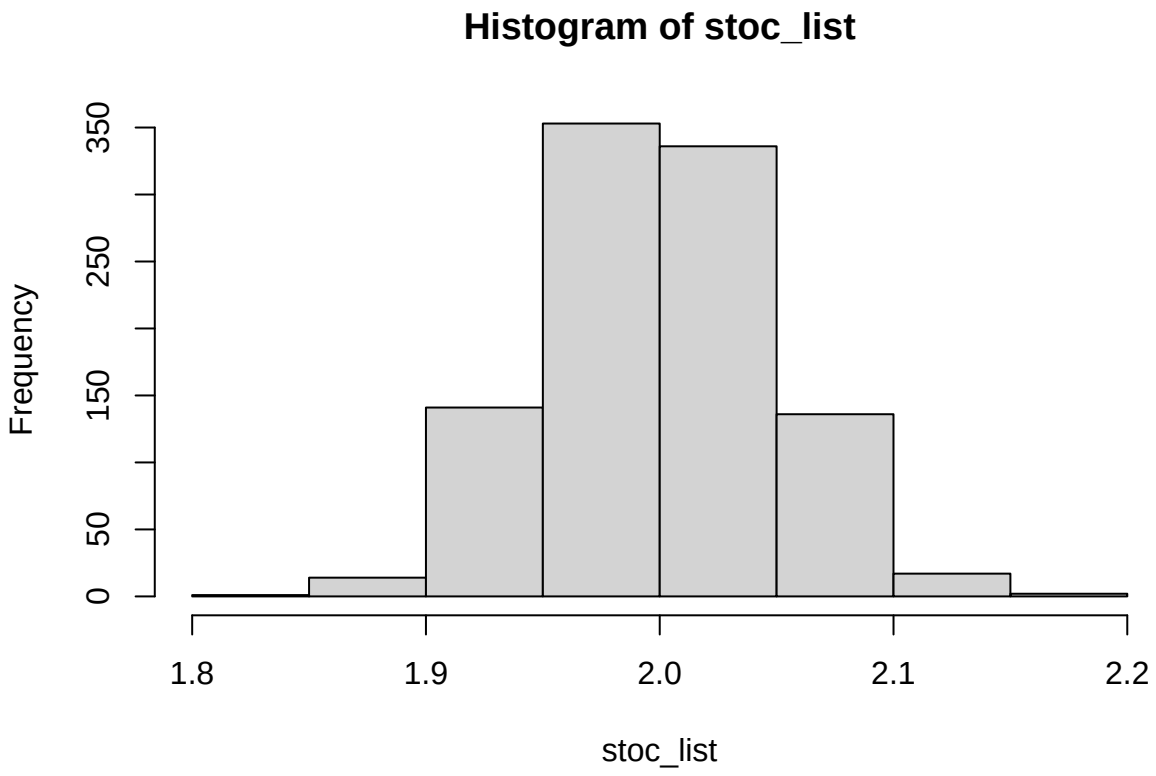
```
p1 = hist(ana_list)
```




```
p2 = hist(batch_list)
```

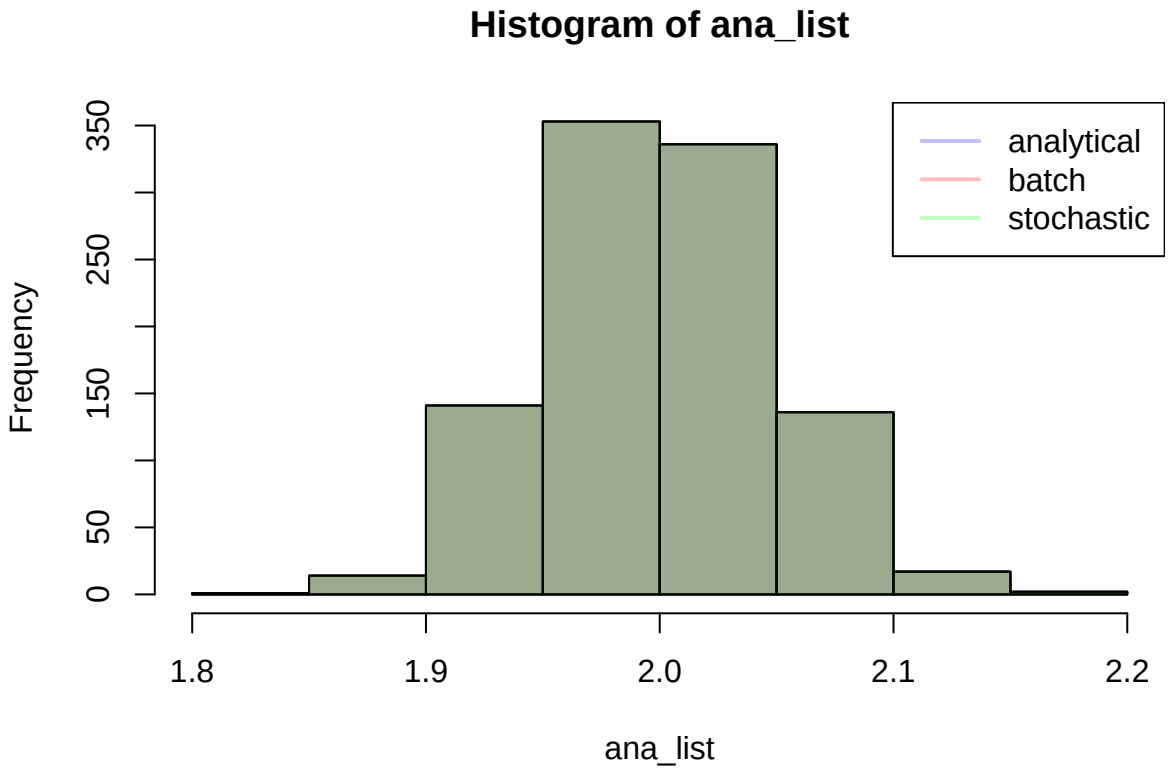



```
p3 = hist(stoc_list)
```




```
plot( p1, col=rgb(0,0,1,1/4))  # first histogram
plot( p2, col=rgb(1,0,0,1/4), add=T)
plot( p3, col=rgb(0,1,0,1/4), add=T)
```

```
legend("topright",         # Add legend to plot
       c("analytical", "batch", "stochastic"),
       col = (c(rgb(0,0,1,1/4),rgb(1,0,0,1/4),rgb(0,1,0,1/4))),
       lty = 1,lwd=2)
```



Q. Comment on how the choice of the algorithm affects the estimates of the slope parameter. Ans 4b: As we can see the overlay,there is an overlap of all three histograms.If we consider analytical solution as the ground truth then we can notice that batch and stochastic gradient descent successfully produce the same results with almost same frequency for the given input.

```
#Ans 4c
set.seed(1)
x = matrix(runif(50,-20,20))
e = rnorm(50,0,4)
X =  cbind(1,x)
y = 3 + 2 * x +e


print("Output of Analytical Solution :")
```

```
## [1] "Output of Analytical Solution :"
```

```
para1 = analytical_solution(X,y)
print(paste("Optimal Intercept: ",para1[1][1],"Optimal slope: ",para1[2][1]))
```

```
## [1] "Optimal Intercept:  3.38644304659435 Optimal slope:  2.00677892557822"
```

```
print("Output of gradient descent(MAE) :")
```

```
## [1] "Output of gradient descent(MAE) :"
```

```
para2 = gradient_descent(x,y,0.01,1000,"mean_absolute")
print(paste("Optimal Intercept: ",para2[2][1],"Optimal slope: ",para2[1][1]))
```

```
## [1] "Optimal Intercept:  2.80475953884644 Optimal slope:  2.00302081130816"
```

```
print("Output of gradient descent(Huber) :")
```

```
## [1] "Output of gradient descent(Huber) :"
```

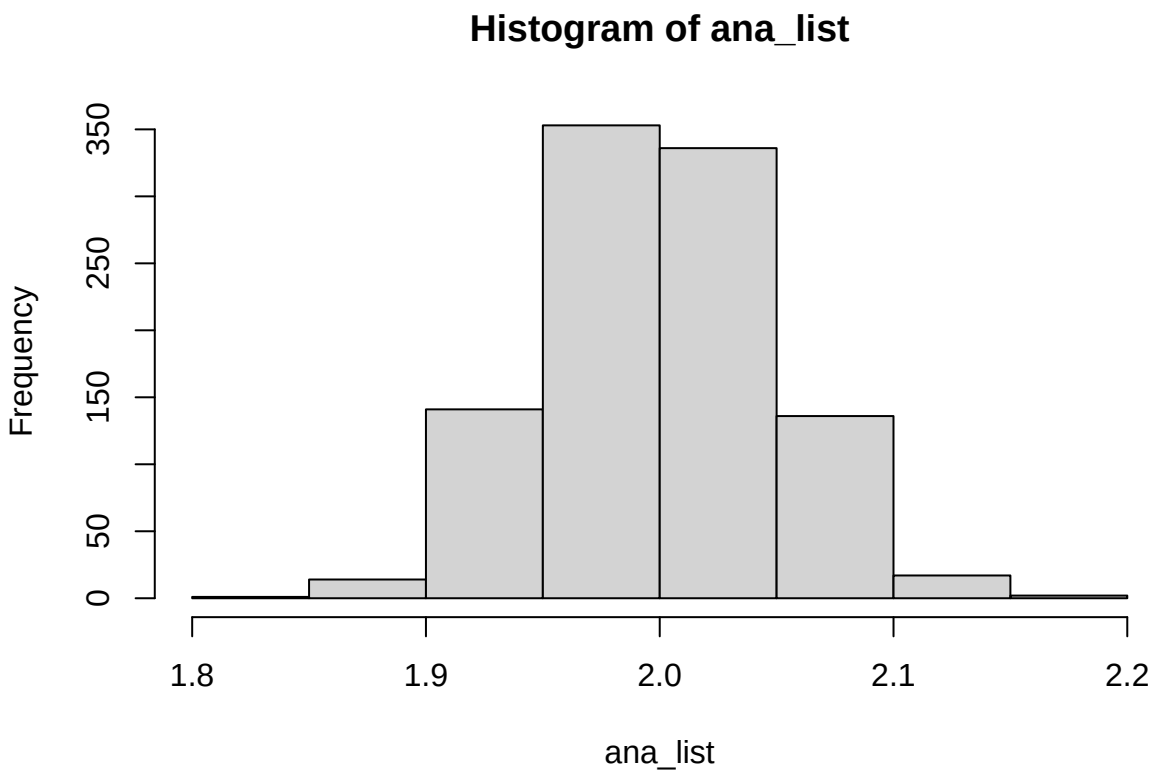
```
para3 = gradient_descent(x,y,0.01,1000,"huber")
print(paste("Optimal Intercept: ",para3[2][1],"Optimal slope: ",para3[1][1]))
```

```
## [1] "Optimal Intercept:  3.22996727735752 Optimal slope:  1.99506149386343"
```

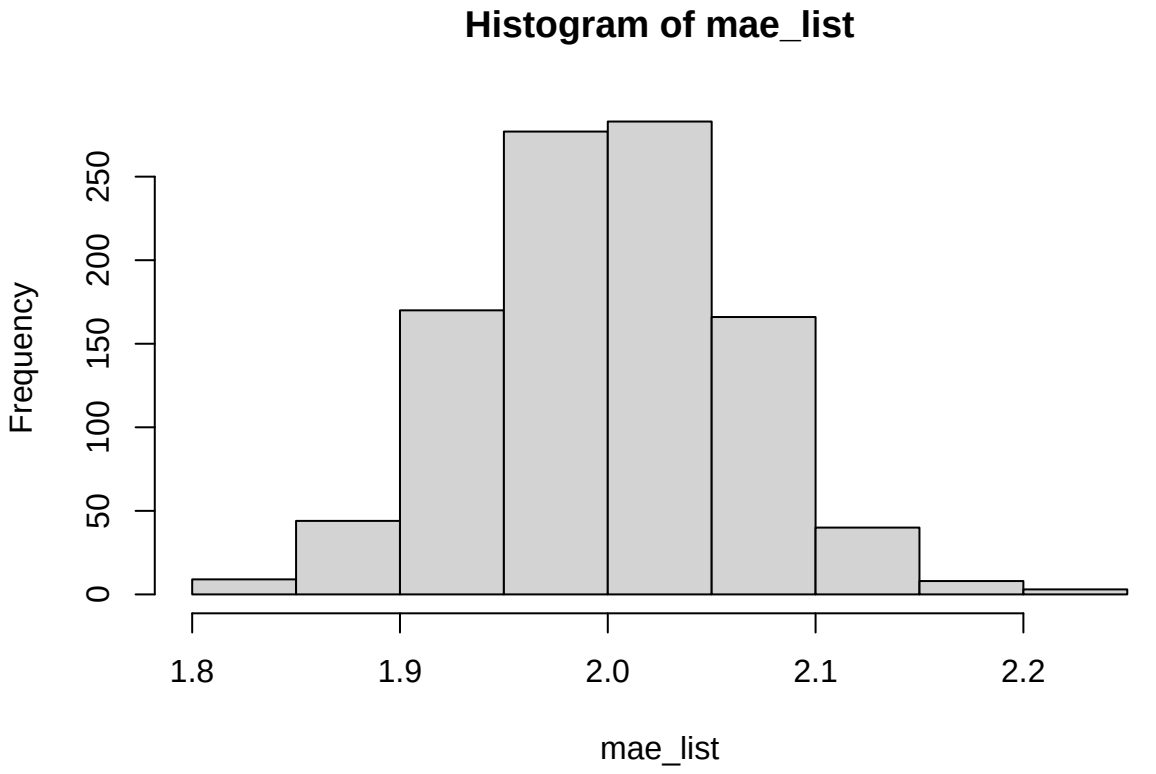
```
#Ans 4d

ana_list = c()
mae_list = c()
huber_list = c()
for (i in 1:1000){
  x = matrix(runif(50,-20,20))
  e = rnorm(50,0,4)
  y = 3 + 2 * x + e
  X = cbind(1,x)
  ana_list = append(ana_list,analytical_solution(X,y)[2])
  mae_list = append(mae_list,gradient_descent(x,y,0.01,1000,"mean_absolute")[1])
  huber_list = append(huber_list,gradient_descent(x,y,0.01,1000,"huber")[1])
}
print("done")
```

```
## [1] "done"
```

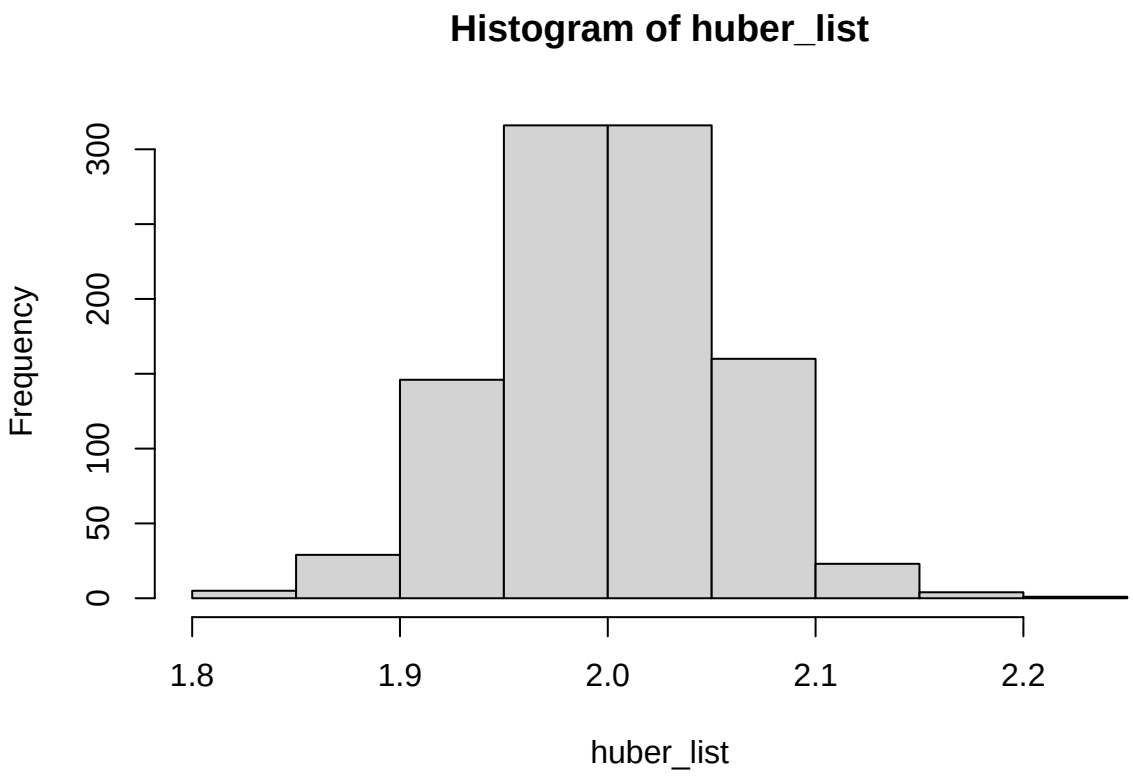
```
p1 = hist(ana_list)
```



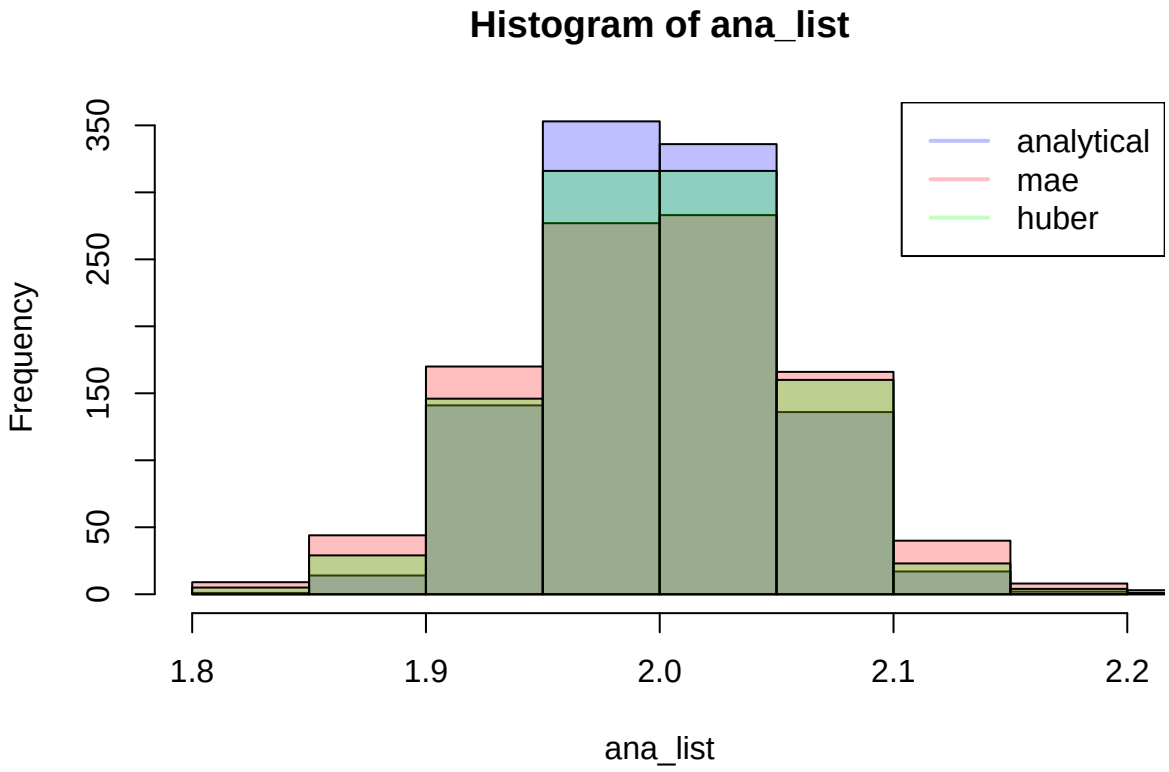

```
p2 = hist(mae_list)
```




```
p3 = hist(huber_list)
```




```
plot( p1, col=rgb(0,0,1,1/4))  # first histogram
plot( p2, col=rgb(1,0,0,1/4), add=T)
plot( p3, col=rgb(0,1,0,1/4), add=T)
legend("topright",         # Add legend to plot
        c("analytical", "mae", "huber"),
        col = (c(rgb(0,0,1,1/4),rgb(1,0,0,1/4),rgb(0,1,0,1/4))),
        lty = 1,lwd=2)
```



Q. Comment on how the choice of the loss function in the case of Normal distribution affects the estimates of the slope parameter.

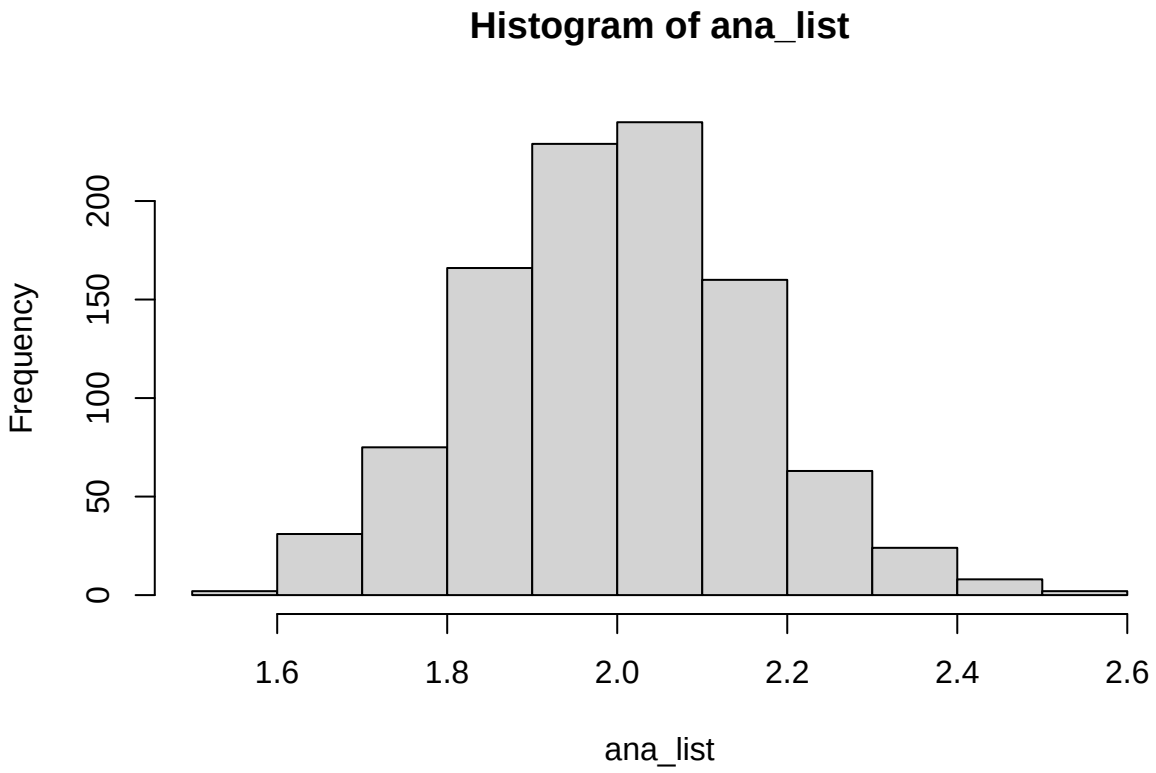
Ans 4e: Consider Analytical Solution as the ground truth we can notice that the rank of frequency for values close to 2(slope = 2) is first being Analytical solution,second Huber and third mean absolute error.In this case we can say that Huber performs better than MAE since its frequency of output is close to the slope(slope =2).

```
# ans 4e
double_val<- function(x){
  y = 2*x
  return(y)
}
half_val <- function(x){
  y = x/2
  return(y)
}

ana_list = c()
mae_list = c()
huber_list = c()
for (i in 1:1000){
  x = matrix(runif(50,-20,20))
  e = rnorm(50,0,4)
  y = 3 + 2 * x + e
  X = cbind(1,x)
  sel_ten = sample(1:50,10)
  sel_pos = sample(sel_ten,5)
  sel_neg = setdiff(sel_ten,sel_pos)
  y[sel_pos] = invisible(lapply(y[sel_pos],double_val))
  y[sel_neg] = invisible(lapply(y[sel_pos],half_val))
```

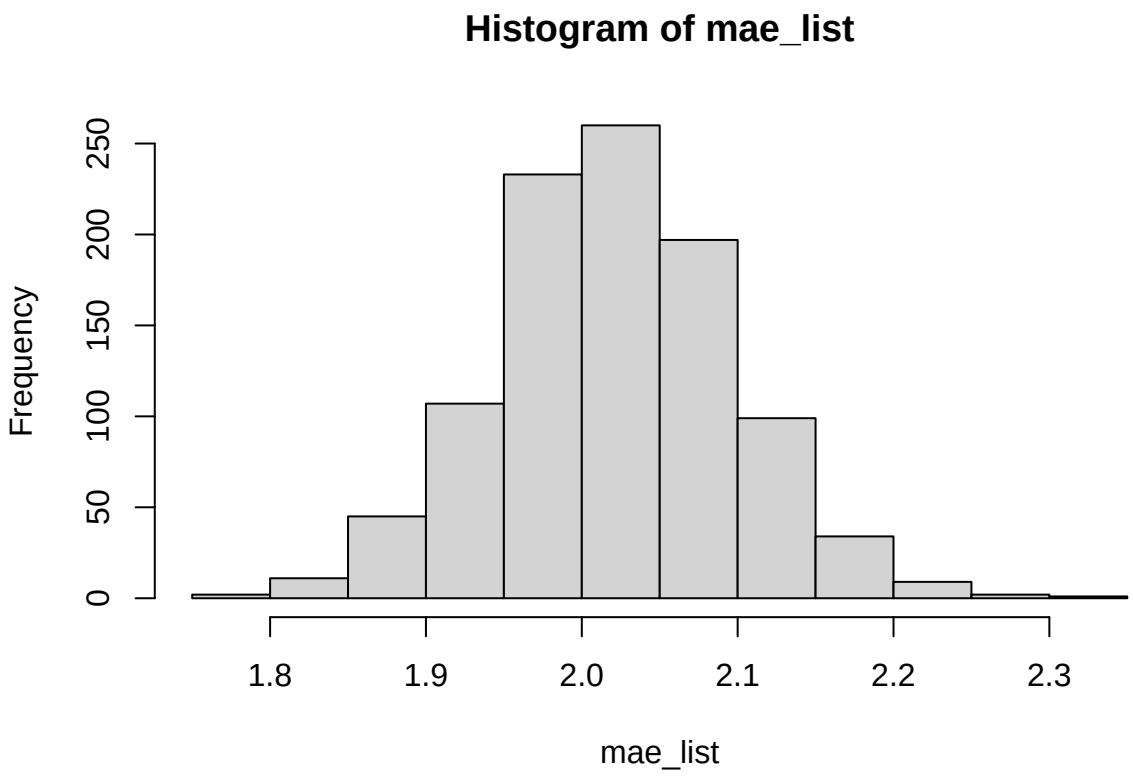
```
  y = matrix(unlist(y))
  ana_list = append(ana_list,analytical_solution(X,y)[2])
  mae_list = append(mae_list,gradient_descent(x,y,0.01,1000,"mean_absolute")[1])
  huber_list = append(huber_list,gradient_descent(x,y,0.01,1000,"huber")[1])

}
print("done")
```

```
## [1] "done"
```

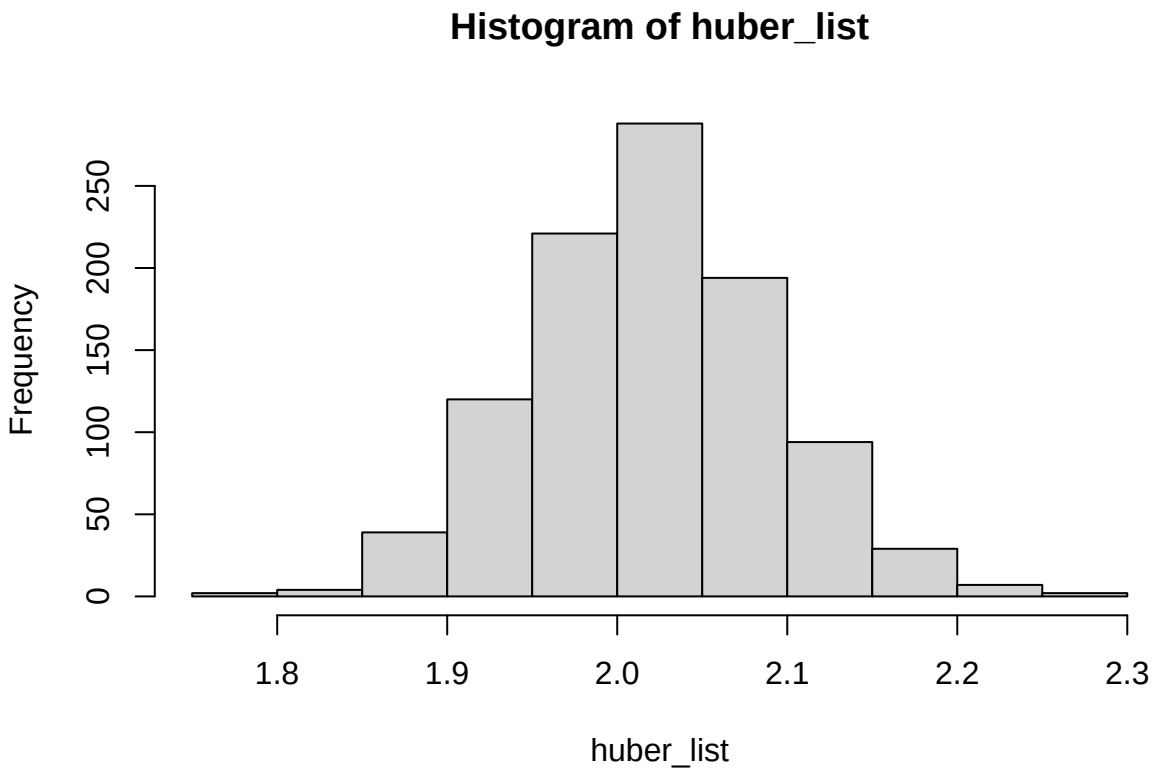
```
p1 = hist(ana_list)
```




```
p2 = hist(mae_list)
```

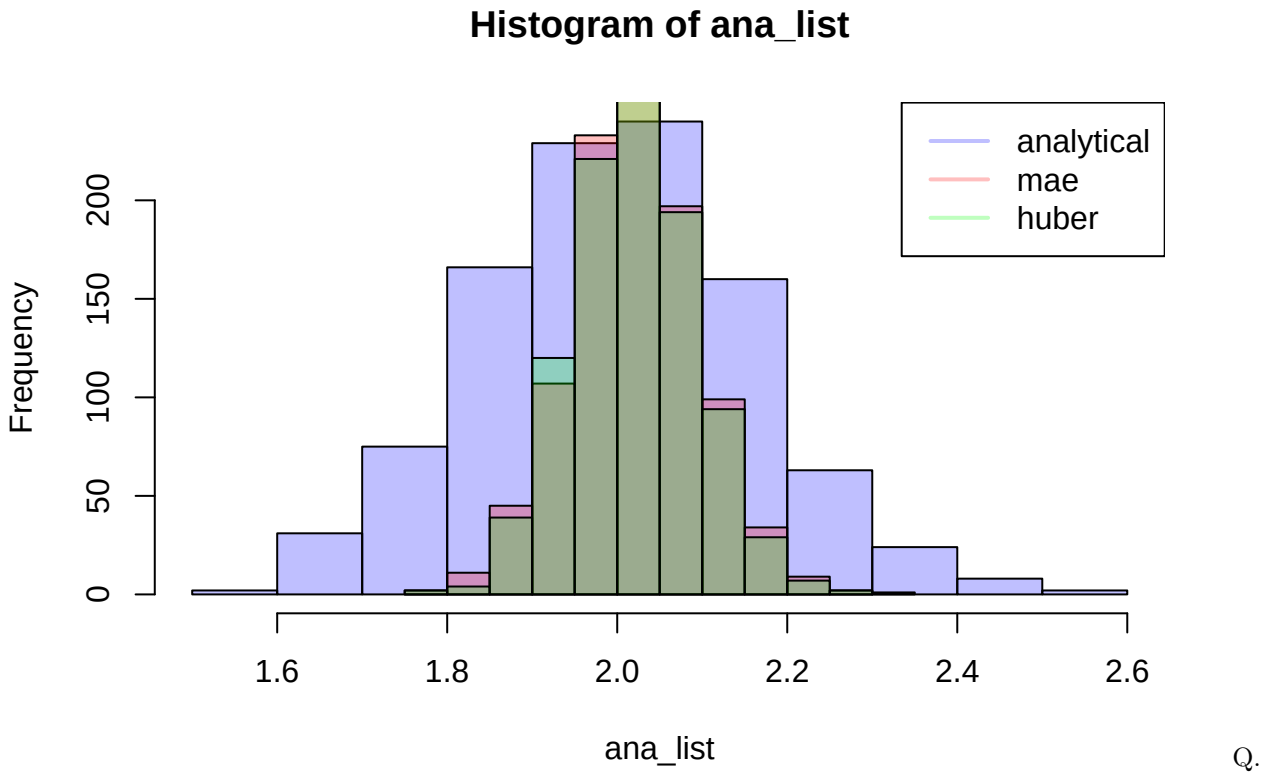



```
p3 = hist(huber_list)
```




```
plot( p1, col=rgb(0,0,1,1/4))  # first histogram
plot( p2, col=rgb(1,0,0,1/4), add=T)
plot( p3, col=rgb(0,1,0,1/4), add=T)
```

```
legend("topright",         # Add legend to plot
       c("analytical", "mae", "huber"),
       col = (c(rgb(0,0,1,1/4),rgb(1,0,0,1/4),rgb(0,1,0,1/4))),
       lty = 1,lwd=2)
```



Q. Comment on how the choice of the loss function in presence of outliers affects the estimates of the slope parameter.

Ans 4f: After introducing outliers in y,we can notice that the high output frequency is still concentrated near the slope (slope = 2).Consider Analytical Solution as the ground truth, we can say that Huber as well as MAE both performed well in this case.Since the outliers introduced didn't skew the data,MAE and Huber both performed equally well in this case.But to choose one huber performed a bit better than mae.